UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/089,139	08/19/2002	Adam Bosworth	109870-130097	2275

25943   7590   02/21/2008
SCHWABE, WILLIAMSON & WYATT, P.C.
PACWEST CENTER, SUITE 1900
1211 SW FIFTH AVENUE
PORTLAND, OR 97204

EXAMINER
RAMPURIA, SATISH

ART UNIT	PAPER NUMBER
2191	

MAIL DATE	DELIVERY MODE
02/21/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

PTOL-90A (Rev. 04/07)

MAILED

FEB 2 1 2008

Technology Center 2100

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Application Number: 10/089,139
Filing Date: August 19, 2002
Appellant(s): BOSWORTH ET AL.

Robert C. Peck

For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 12/04/2007 appealing from the Office action

mailed 03/22/2007.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

The following are the related appeals, interferences, and judicial proceedings known to the examiner which may be related to, directly affect or be directly affected by or have bearing on the board's decision in the pending appeal:

The Real Party in Interest is BEA Systems, Inc., of 2315 North First Street, San Jose, California 95131, assignee of the application. Assignment of the application from the Inventors to BEA Systems, Inc. is recorded with the United States Patent and Trademark Office on August 19, 2002, at Reel 013198 Frame 0130.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

The amendment after final rejection filed on 3/22/2007.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6,292,936	Wang	09-2001
6,732,330	Claussen et al.	5/4/2004

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless --
(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

Claims 1, 2, 3, 6, 7, 20, 21, 22, 25, and 26 are rejected under 35 U.S.C. 102(e) as being anticipated by 6,292,936 to Wang (hereinafter, Wang).

Per claims 1 and 2:

Wang discloses:

- A method of computing comprising:

- reading, by an execution engine (col. 2, lines 49-55 "...run time processors...that are executed...comprises a Java Virtual Machine...executes Java programming statements...VisualBasic Script interpreter... executes VisualBasic Script programming statements"), a data processing representation having code sections with code statements of at least a first and a second programming language (col. 1, lines 44-46 "Each of the runtime processors processes their respective corresponding intermediate sources derived from an original input source in a synchronous manner");

- recognizing, by an execution engine (col. 2, lines 49-55 "...run time processors...that are executed...comprises a Java Virtual Machine...executes Java programming statements...VisualBasic Script interpreter... executes VisualBasic Script programming statements"), a

first code section with at least code statements of a first programming

language (col. 2, lines 56-59 "The server system 106 may further include

one or more translators 114 that are executed to translate the original

input source for the runtime processors 110 and 112");

- invoking, by an execution engine (col. 2, lines 49-55 "...run time

processors...that are executed...comprises a Java Virtual

Machine...executes Java programming statements...VisualBasic Script

interpreter... executes VisualBasic Script programming statements"), a

first code statement processing unit of the first programming language to

process the first code section (col. 1, lines 44-46 "Each of the runtime

processors processes their respective corresponding intermediate sources

derived from an original input source in a synchronous manner");

- recognizing, by an execution engine (col. 2, lines 49-55 "...run time

processors...that are executed...comprises a Java Virtual

Machine...executes Java programming statements...VisualBasic Script

interpreter... executes VisualBasic Script programming statements"), a

second code section with at least code statements of a second

programming language (col. 1, lines 46-48 "One or more of the respective

corresponding intermediate sources includes a synchronizer token that

provides synchronization among the runtime processors");

- invoking, by an execution engine (col. 2, lines 49-55 "...run time

processors...that are executed...comprises a Java Virtual

Machine...executes Java programming statements...VisualBasic Script
interpreter... executes VisualBasic Script programming statements"), a
second code statement processing unit of the second programming
language to process the second code section (col. 1, lines 49-51 "Using
the synchronizer token, an execution sequence of the original input source
is maintained").

Per claim 3:

The rejection of claim 1 is incorporated, and further, Wang discloses:

-   wherein said second code section is embedded within said first code
    section. The limitations in the claims are similar to those in claim 1, and
    rejected under the same rational set forth in connection with the rejection
    of claim 1.

Per claim 6:

The rejection of claim 1 is incorporated, and further, Wang discloses:

-   recognizing a third code section with at least code statements of a third
    programming language (col. 2, lines 56-59 "The server system 106 may
    further include one or more translators 114 that are executed to translate
    the original input source for the runtime processors 110 and 112");
-   invoking a third code statement processing unit of the third programming
    language to process the third code section (col. 1, lines 44-46 "Each of the

runtime processors processes their respective corresponding intermediate

sources derived from an original input source in a synchronous manner").

Per claim 7:

The rejection of claim 6 is incorporated, and further, Wang discloses:

- wherein said third code section is embedded within said second code

  section, and said second code section is embedded within said first code

  section. The limitations in the claims are similar to those in claim 6, and

  rejected under the same rational set forth in connection with the rejection

  of claim 6.

Claims 20, 21, 22, 25 and 26 are the apparatus claim corresponding to method claims

1, 3, 6 and 7 respectively, and rejected under the same rational set forth in connection

with the rejection of claims 1, 3, 6 and 7 respectively, above, as noted above and Wang

also discloses system, see FIG. 1 and associated text.

Claim Rejections - 35 USC § 103
The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section
102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the
subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary
skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the
invention was made.

Claims 4, 5, 8, 23, 24 and 27 are rejected under 35 U.S.C. 103(a) as being

unpatentable over Wang in view of US Patent No. 6,732,330 to Claussen et al.

(hereinafter, Claussen).

Per claim 4:

The rejection of claim 1 is incorporated, and further, Wang does not explicitly disclose

wherein said first language is a directive language, and said second language is a

selected one of XML and an object-oriented language.

However, Claussen discloses in an analogous computer system wherein said

first language is a directive language, and said second language is a selected one of

XML and an object-oriented language (col. 2-3, lines 66-67 and 1-2 "…supporting

multiple languages is compiled in to an XML… and thereafter, into a Java™ servlet…").

Therefore, it would have been obvious to a person of ordinary skill in the art at

the time the invention was made to incorporate the method of wherein said first

language is a directive language, and said second language is a selected one of XML

and an object-oriented language as taught by Claussen into the method of enabling

multiple runtime processors in an embedded scripting system as taught by Wang. The

modification would be obvious because of one of ordinary skill in the art would be

motivated to use XML and Java to provide a technique for publishing Internet content

that can fully leverage the manipulation as suggested by Claussen (col. 2, lines 23-55).

Per claim 5:

- wherein said first language is an object-oriented language, and said second language is XML. The limitations in the claims are similar to those in claim 4, and rejected under the same rational set forth in connection with the rejection of claim 4.

Per claim 8:

- wherein said first language is a directive language, said second language is an object-oriented language and said third language is XML. The limitations in the claims are similar to those in claim 4, and rejected under the same rational set forth in connection with the rejection of claim 4.

Claims 23, 24 and 27 are the apparatus claim corresponding to method claims 4, 5 and 8 respectively, and rejected under the same rational set forth in connection with the rejection of claims 4, 5 and 8 respectively, above, as noted above and Wang also discloses system, see FIG. 1 and associated text.

Claims 9-13 and 28-32 are rejected under 35 U.S.C. 103(a) as being unpatentable over Wang in view of US Patent No. 5,428,792 to Conner et al. (hereinafter, Conner).

Per claim 9:

The rejection of claim 1 is incorporated, and further, Wang discloses:

- invoking the library function, and outputting the result of the invocation

(col. 3, lines 40-42 "The remaining VisualBasic Script blocks in the original

input source 116 are translated into notify method and wait method

invocations").

Wang does not explicitly disclose wherein the method further comprises recognizing an

invocation of a library function within at least a selected one of said first and second

code sections.

However, Conner discloses in an analogous computer system wherein the

method further comprises recognizing an invocation of a library function within at least a

selected one of said first and second code sections (col. 7, lines 20-23 "class designer

defines the class interface, implements the class methods, and finally loads the

resulting object code into a class library").

Therefore, it would have been obvious to a person of ordinary skill in the art at

the time the invention was made to incorporate the method of recognizing an invocation

of a library function within at least a selected one of said first and second code sections

as taught by Conner into the method of enabling multiple runtime processors in an

embedded scripting system as taught by Wang. The modification would be obvious

because of one of ordinary skill in the art would be motivated to use a library function to

provide the reusability of the OOP functions already exist as suggested by Conner (col.

1, lines 55-67).

Per claim 10:

The rejection of claim 1 is incorporated, and further, Wang does not explicitly disclose wherein the library function is a selected one of an emit function for outputting execution results, a pop function for returning an element, and a push function for backing up an insertion point.

However, Conner discloses in an analogous computer system wherein the library function is a selected one of an emit function for outputting execution results, a pop function for returning an element, and a push function for backing up an insertion point (col. 5, lines 1-12 "...class is a definition of an object... <stack> is an example of a class... stack contains two data elements (<stackArray> and <stackTop>), and supports three methods, <create()>, <push()>, and <pop()>...").

The feature of library function is a selected one of an emit function for outputting execution results, a pop function for returning an element, and a push function for backing up an insertion point would be obvious for the reasons set forth in the rejection of claim 9.

Per claim 11:

The rejection of claim 1 is incorporated, and further, Wang does not explicitly disclose wherein the method further comprises recognizing a header section of a selected one of the first and the second programming; recognizing a directive statement within the header section, enumerate one or more data packages; and importing the enumerated

one or more data packages for use within code sections with at least statements of the selected first and second programming language.

However, Conner discloses in an analogous computer system wherein the method further comprises recognizing a header section of a selected one of the first and the second programming language (col. 9, lines 35-40 "...a valid C header file which contains macros necessary to invoke public methods and access public data elements of the class... file... included in any client of the class, and is created by the SOM compiler"); recognizing a directive statement within the header section, enumerate one or more data packages (col. 25, lines 14-20 "section contains an include statement that is a directive to the OIDL preprocessor telling the compiler where to find the class interface definition for this class' parent class..."); and importing the enumerated one or more data packages for use within code sections with at least statements of the selected first and second programming language (col. 2, lines 19-21 "...bindings are input to the particular target language compiler to generate object module...").

The feature of recognizing a header section... recognizing a directive statement... and importing the enumerated... would be obvious for the reasons set forth in the rejection of claim 9.

Per claim 12:

- wherein the method further comprises recognizing a header section of a selected one of the first and the second programming language;

recognizing a declare statement within the header section, enumerating

one or more processing methods; and instantiating the enumerated one or

more processing methods for use within code sections with at least

statements of the selected first and second programming language. The

limitations in the claims are similar to those in claim 11, and rejected under

the same rational set forth in connection with the rejection of claim 11.

Per claim 13:

- wherein the method further comprises recognizing a header section of a

    selected one of the first and the second programming language;

    recognizing a declare statement within the header section, enumerating

    one or more instance variables; and instantiating the enumerated one or

    more instance variables for use within code sections with at least

    statements of the selected first and second programming language. The

    limitations in the claims are similar to those in claim 11, and rejected under

    the same rational set forth in connection with the rejection of claim 11.

Claims 28-32 are the apparatus claim corresponding to method claims 9-13

respectively, and rejected under the same rational set forth in connection with the

rejection of claims 9-13 respectively, above, as noted above and Wang also discloses

system, see FIG. 1 and associated text.

(10) Response to Argument

ARGUMENTS (pages 5-10),

Appellant argued that:

(i) Appellants respectfully disagree with the Examiner's rejection. No specific
reasons are provided for the rejection of claims 20-32 under §112, second paragraph.
The only specific reason given is only mentioned in regard to claims 14-19 and 33-38,
which were cancelled in Appellant's Response to the Final Office Action, mailed April
25, 2007. Further, the reasons given for rejecting claims 14-19 and 33-38 do not apply
to claims 20-32. Claims 14-19 and 33-38 were rejected for reciting "adapted to" and
"first and second code statements." Claims 20-32 do not include these recitations. (Page 5).

Examiner's Response

In response to Appellant arguments with respect to 112 second paragraph

rejection to claims 20-32 appears to be a typo. The claims 20-32 were listed only in the

heading, however, no specific rejection/reasons were given as the claims do not recite

the limitations as cited in the claims which were given the specific reasons for rejection.

Therefore, the rejection to claims 20-32 were not tend to be rejected under 112 second

paragraph.

Appellant argued that:

(ii) In contrast, Wang fails to disclose, expressly or inherently, an execution engine that invokes
first **and** second code statement processing units of first and second programming languages, as is
claimed in amended claim 1. Wang merely teaches "an interpreter-based scripting environment [that]
includes multiple runtime processors executed by the computer. Each of the runtime processors
processes their respective corresponding intermediate sources derived from an original source in a
synchronous manner" (abstract). The processors are interdependently invoked. Specifically, the original
source disclosed in Wang comprises an HTML document with embedded Visual Basic scripting language
blocks. Wang teaches a single HTML parser that parses the original HTML + VB source, and translates
the non-VB source into a first intermediate source executable by a Java VM, and the VB source into a
second intermediate source having the VB script statements executable by a VB script interpreter.

Even if we were to read the Java VM and VB Script Interpreter as the recited first and second
code statement processing units, Wang does not disclose an execution engine that invokes both of the
Java VM and VB Script Interpreter. The HTML parser of Wang, described above, simply creates

intermediate sources and does not invoke either of the Java VM or the VB Script Interpreter. According to Wang, col. 3, lines 57- 67 and col. 4, lines 1-8, the Java VM is invoked first at runtime, and the VB Script Interpreter is later invoked by the Java VM. Thus, there is no common execution engine that invokes both the Java VM and the VB Script Interpreter and, therefore, Wang does not disclose the execution engine recited by claim 1. (Page 7).

Examiner's Response

In response to Appellant arguments Wang discloses enabling multiple runtime processor executed by the computer. Each of the runtime processors process their respective intermediate sources derived from an original input source, i.e., Java or Visual Basic Script (col. 1, lines 37-51). In order to process multi language processor, Wang's system recognizes different input source languages and invokes the respective processor according to the input source language (col. 2, lines 26-35). More particularly, Wang discloses during runtime Java Virtual machine and VisualBasic Script interpreter execute their respective corresponding sources. Java Virtual machine performs normal processing of intermediate source until it invokes a thread object run method to initiate (invoke) the execution of VisualBasic Script interpreter (col. 3, lines 50-67). Then once invoked, the VisualBasic Script interpreter performs its normal processing of the intermediate source (col. 4, lines 1-5). Thus, Wang's system discloses that first programming language (Java) and second programming language (VisualBasic) are executing by their respective processors as claimed in claim 1. For the 'common execution engine', again, examiner does not find these limitations either in the claims or in the Applicants specification. However, for the sake of the arguments, Wang discloses a runtime processor which invokes an appropriate execution engine i.e., for Java programming statements, Wang's system invokes Java virtual machine that executes

Java programming statements and for VisualBasic Script programming statements,

Wang's system invokes VisualBasic Script interpreter that executes VisualBasic Script

programming statements (col. 2, lines 49-55). Therefore the runtime processor of Wang

is common execution engine for Java and VisualBasic Script. Applicant's argument that

the references fail to show certain features of applicant's invention, it is noted that the

features upon which applicant relies (i.e., common execution engine) are not recited in

the rejected claim(s). Although the claims are interpreted in light of the specification,

limitations from the specification are not read into the claims. See *In re Van Geuns*, 988

F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

Appellant argued that:

(iii) Additionally, Wang fails to clearly and expressly disclose the limitations of claim6. Claim 6 recites recognizing a third code section of a third language and invoking a third code statement processing unit of the third language to process the third section. Wang, in contrast, simply teaches two runtime processors (110 and 112) invoked to process intermediate sources derived from HTML and VB Script. No disclosure is made of a third source of a third language or a third runtime processor. Accordingly, for at least this additional reason, Wang fails to anticipate claim 6. Claims 7 and 8 depend from claim 6 and accordingly are also patentable for at least this additional reason. (Page 9).

Examiner's Response

In response to Appellant arguments Wang discloses enabling multiple runtime

processor executed by the computer. Each of the runtime processors process their

respective intermediate sources derived from an original input source, i.e., Java or

Visual Basic Script (See summary). In order to process multi language processor,

Wang's system recognizes different input source languages and invokes the respective

processor according to the input source language (col. 2, lines 26-35). Further Wang

discloses that the invention is not limited to specific programming languages, and could

comprise languages other than HTML, Java, VisualBasic Script, for example, C, C++,

Perl, Cobol etc. (col. 6, lines 21-25).

Appellant did not present any arguments related to 103(a) rejection to claims 4,

5, 8, 23, 24, 27, 9-13 and 28-32. Therefore, no response or explanation is given.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the

Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

Satish S. Rampuria

Conferees:

Eddie Lee

Wei Zhen (SPE)

Satish S. Rampuria (Examiner)

/Satish S. Rampuria/
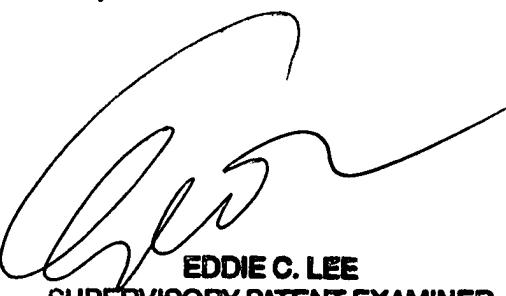
EDDIE C. LEE
SUPERVISORY PATENT EXAMINER
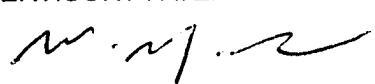
WEI ZHEN
SUPERVISORY PATENT EXAMINER